Express Mail Label No. Dated: _____

Docket No.: 04970/000N023-US0
(PATENT)

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re Patent Application of:
Akishige Yamamoto, et al.

Application No.: 10/628,133 Confirmation No.: 4516

Filed: July 25, 2003 Art Unit: N/A

For: FUNCTION EXECUTION METHOD, FUNCTION EXECUTION APPARATUS, COMPUTER PROGRAM AND RECORDED MEDIUM

Examiner: Not Yet Assigned

CLAIM FOR PRIORITY AND SUBMISSION OF DOCUMENTS

Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

Dear Sir:

Applicant hereby claims priority under 35 U.S.C. 119 based on the following prior foreign application filed in the following foreign country on the date indicated:

Country	Application No.	Date
Japan	2001-088850	March 26, 2001

In support of this claim, a certified copy of the said original foreign application is filed herewith.

Dated: December 12, 2003

Respectfully submitted,

By _____ MARIE GILFILLAN 44085

Joseph B. Lerch
Registration No.: 26,936
DARBY & DARBY P.C.
P.O. Box 5257
New York, New York 10150-5257
(212) 527-7700
(212) 753-6237 (Fax)
Attorneys/Agents For Applicant

日本国特許庁
JAPAN PATENT OFFICE

別紙添付の書類に記載されている事項は下記の出願書類に記載されている事項と同一であることを証明する。

This is to certify that the annexed is a true copy of the following application as filed with this Office.

出願年月日
Date of Application: ２００１年　３月２６日

出願番号
Application Number: 特願２００１－０８８８５０

[ST. 10/C]: ［ＪＰ２００１－０８８８５０］

出願人
Applicant(s): 関西ティー・エル・オー株式会社

２００３年　８月　５日

特許庁長官
Commissioner,
Japan Patent Office 今井康夫

出証番号　出証特２００３－３０６２７５９

【書類名】　特許願
【整理番号】　22039
【提出日】　平成13年 3月26日
【あて先】　特許庁長官殿
【国際特許分類】　G06F 12/02　510
【発明の名称】　関数実行方法、関数実行装置、コンピュータプログラム、及び記録媒体
【請求項の数】　9
【発明者】
【住所又は居所】　東京都新宿区新宿２丁目４番３号　フォーシーズンビル１０階　株式会社数理システム内
【氏名】　山本　晃成
【発明者】
【住所又は居所】　京都府京都市左京区吉田本町　京都大学内
【氏名】　湯淺　太一
【特許出願人】
【識別番号】　899000046
【氏名又は名称】　関西ティー・エル・オー株式会社
【代理人】
【識別番号】　100078868
【弁理士】
【氏名又は名称】　河野　登夫
【電話番号】　06(6944)4141
【手数料の表示】
【予納台帳番号】　001889
【納付金額】　21,000円
【提出物件の目録】
【物件名】　明細書　1
【物件名】　図面　1

【物件名】　要約書　1

【包括委任状番号】　9908225

【プルーフの要否】　要

【書類名】　明細書

【発明の名称】　関数実行方法、関数実行装置、コンピュータプログラム、及び記録媒体

【特許請求の範囲】

【請求項１】　メモリ中のスタック領域に、他の関数を呼び出す処理を含む呼出関数から呼び出される被呼出関数の形式に基づく関数記録領域を積み上げ、積み上げた関数記録領域を利用して被呼出関数を呼び出し、呼び出された被呼出関数の実行後、積み上げた関数記録領域を破棄する関数実行方法において、

関数記録領域で実行される呼出関数を解析し、

該解析により呼出関数が所定の条件を満足すると判断した場合に、呼出関数を実行する前記関数記録領域を、被呼出関数を呼び出す領域として利用する

ことを特徴とする関数実行方法。

【請求項２】　前記所定の条件は、被呼出関数の実行結果が呼出関数の実行結果となることであることを特徴とする請求項１に記載の関数実行方法。

【請求項３】　前記呼出関数及び被呼出関数が異なるときに、被呼出関数の形式に基づいて、前記関数記録領域を変更することを特徴とする請求項１又は請求項２に記載の関数実行方法。

【請求項４】　メモリ中のスタック領域に、他の関数を呼び出す処理を含む呼出関数から呼び出された被呼出関数の形式に基づく関数記録領域を積み上げ、積み上げた関数記録領域を利用して被呼出関数を呼び出し、呼び出された被呼出関数の実行後、積み上げた関数記録領域を破棄する関数実行装置において、

関数記録領域で実行される呼出関数を解析する手段と、

該解析により呼出関数が所定の条件を満足すると判断した場合に、呼出関数を実行する前記関数記録領域を、被呼出関数を呼び出す領域として利用する手段と

を備えることを特徴とする関数実行装置。

【請求項５】　コンピュータに、メモリ中のスタック領域に、他の関数を呼び出す処理を含む呼出関数から呼び出される被呼出関数の形式に基づく関数記録領域を積み上げさせ、積み上げた関数記録領域を利用して被呼出関数を呼び出させ、呼び出された被呼出関数の実行後、積み上げた関数記録領域を破棄させるコ

ンピュータプログラムにおいて、

コンピュータに、関数記録領域で実行される呼出関数を解析させる手順と、

コンピュータに、解析により呼出関数が所定の条件を満足すると判断した場合に、呼出関数を実行する前記関数記録領域を、被呼出関数を呼び出す領域として利用させる手順と

を含むことを特徴とするコンピュータプログラム。

【請求項６】　前記所定の条件は、被呼出関数の実行結果が呼出関数の実行結果となることであることを特徴とする請求項５に記載のコンピュータプログラム。

【請求項７】　前記呼出関数及び被呼出関数が異なるときに、被呼出関数の形式に基づいて、前記関数記録領域を変更することを特徴とする請求項５又は請求項６に記載のコンピュータプログラム。

【請求項８】　コンピュータに、メモリ中のスタック領域に、他の関数を呼び出す処理を含む呼出関数から呼び出される被呼出関数の形式に基づく関数記録領域を積み上げさせ、積み上げた関数記録領域を利用して被呼出関数を呼び出させ、呼び出された被呼出関数の実行後、積み上げた関数記録領域を破棄させるコンピュータプログラムにおいて、

コンピュータに、関数記録領域で実行される第１呼出関数を解析させる手順と、

コンピュータに、解析により第１呼出関数が所定の条件を満足すると判断した場合に、呼出関数を実行する前記関数記録領域を、被呼出関数を呼び出す領域として利用させる処理を含む第１と異なる第２呼出関数を、第１呼出関数の代替関数として実行させる手順と

を含むことを特徴とするコンピュータプログラム。

【請求項９】　コンピュータに、メモリ中のスタック領域に、他の関数を呼び出す処理を含む呼出関数から呼び出される被呼出関数の形式に基づく関数記録領域を積み上げさせ、積み上げた関数記録領域を利用して被呼出関数を呼び出させ、呼び出された被呼出関数の実行後、積み上げた関数記録領域を破棄させるコンピュータプログラムを記録してある、コンピュータでの読み取りが可能な記録

媒体において、

コンピュータに、関数記録領域で実行される呼出関数を解析させる手順と、

コンピュータに、解析により呼出関数が所定の条件を満足すると判断した場合に、呼出関数を実行する前記関数記録領域を、被呼出関数を呼び出す領域として利用させる手順と、

を含むコンピュータプログラムを記録してあることを特徴とするコンピュータでの読み取りが可能な記録媒体。

【発明の詳細な説明】

【０００１】

【発明の属する技術分野】

メモリ中のスタック領域に、他の関数を呼び出す処理を含む呼出関数から呼び出される被呼出関数の形式に基づく関数記録領域を積み上げ、積み上げた関数記録領域を利用して被呼出関数を呼び出し、呼び出された被呼出関数の実行後、当該関数領域を破棄する関数実行方法、その方法を適用した関数実行装置、その装置を実現するためのコンピュータプログラム、及びそのコンピュータプログラムを記録した記録媒体に関し、特にＪＶＭ(Java Virtual Machine)にてＪａｖａ言語等のオブジェクト指向の言語にて記述された関数を実行する関数実行方法、関数実行装置、コンピュータプログラム、及び記録媒体に関する。

【０００２】

【従来の技術】

Ｊａｖａ言語等のオブジェクト指向の言語にて記述された複数の命令からなる関数を複数含むプログラムを実行する関数実行方法が近年様々な用途で利用されており、またこのようなＪａｖａ言語は一般的にＪＶＭのバイトコードにコンパイルされてからＪＶＭのメモリ中の領域にて実行される。

【０００３】

図７はプログラムの実行に必要な関数を記録する関数記録領域を示す概念図である。

プログラムは、一般に関数（メソッド）の呼び出しが入れ子状態になって実行される。即ち図７（ａ），（ｂ），（ｃ）に示すようにプログラムの処理に必要

な関数は、実行中の呼出関数から実行すべき被呼出関数を呼び出して、メモリ中のスタック領域に確保された実行中の呼出関数の関数記録領域（フレーム）に、被呼出関数の形式に基づく新たな関数記録領域を積み上げる形で確保し、確保された関数記録領域を利用して呼び出された被呼出関数を実行し、被呼出関数の実行完了後、積み上げられている不要になった関数記録領域を破棄する。

【０００４】

最上位に積み上げられた実行中の関数を記録している関数記録領域は、トップフレームと呼ばれ、図７（ａ）では関数Ｆが実行中であり、関数Ｆの関数記録領域がトップフレームとなる。

この関数Ｆを呼出関数として、被呼出関数である関数Ｇが呼び出されると、図７（ｂ）に示すように、関数Ｇを実行するための関数記録領域がスタック領域に積み上げられてトップフレームとなり、関数Ｇの実行完了後、関数Ｇの実行結果を関数Ｆに渡し、当該関数記録領域は破棄されて、呼び出し前の図７（ａ）の状態となる。

【０００５】

なお図７（ｂ）において、関数Ｇを呼出関数として、更に他の被呼出関数である関数Ｈが呼び出されたときは、図７（ｃ）に示すように、関数Ｈを実行するための関数記録領域がスタック領域に積み上げられてトップフレームとなり、関数Ｈの実行完了後、関数Ｈの実行結果を関数Ｇに渡し、当該関数記録領域は破棄されて、呼び出し前の図７（ｂ）の状態となる。

【０００６】

このとき呼出関数である関数Ｇにより呼び出された被呼出関数である関数Ｈの実行結果を関数Ｇの実行結果として関数Ｆに渡す場合、関数Ｈの呼び出しを末尾呼び出しと呼び、末尾呼び出しでは関数Ｈの実行完了後、関数Ｈ及び関数Ｇの関数記録領域は連続して破棄されることになる。

【０００７】

【発明が解決しようとする課題】

しかしながらスタック領域に数多くの関数記録領域を積み上げていくことにより、スタック領域をオーバーフローするという状況を引き起こし、場合によって

はプログラムの実行に支障を来すことがあるという問題がある。

【０００８】

また関数記録領域の積み上げ及び破棄に要する処理負荷が、全体の実行速度の低下を招くという問題がある。

【０００９】

本発明は斯かる事情に鑑みてなされたものであり、呼出関数による呼び出しが末尾呼び出しであると判断した場合に、新たな関数記録領域を積み上げることなく、呼出関数が記録されている関数記録領域を、被呼出関数を記録する関数記録領域として利用することで、関数記録領域がスタック領域をオーバーフローしてプログラムの実行に支障を来す可能性を低減し、また積み上げ及び破棄に要する処理負荷を低減し、全体の実行速度を向上させる関数実行方法、その方法を適用した関数実行装置、その装置を実現するためのコンピュータプログラム、及びそのコンピュータプログラムを記録した記録媒体の提供を主たる目的とする。

【００１０】

さらに呼出関数及び被呼出関数が同じ場合には、確保されている関数記録領域をそのまま利用し、呼出関数及び被呼出関数が異なる場合には、確保されている関数記録領域を被呼出関数の形式に基づいて変更することにより、全体の処理負荷の軽減及び関数記録領域の適正化を行うことが可能な関数実行方法等の提供を他の目的とする。

【００１１】

【課題を解決するための手段】

第１発明に係る関数実行方法は、メモリ中のスタック領域に、他の関数を呼び出す処理を含む呼出関数から呼び出される被呼出関数の形式に基づく関数記録領域を積み上げ、積み上げた関数記録領域を利用して被呼出関数を呼び出し、呼び出された被呼出関数の実行後、積み上げた関数記録領域を破棄する関数実行方法において、関数記録領域で実行される呼出関数を解析し、該解析により呼出関数が所定の条件を満足すると判断した場合に、呼出関数を実行する前記関数記録領域を、被呼出関数を呼び出す領域として利用することを特徴とする。

【００１２】

第１発明に係る関数実行方法では、呼出関数が所定の条件を満足した場合に、呼出関数の関数記録領域を被呼出関数の記録領域として再利用して、スタック領域に積み上げられる関数記録領域の数を低減することで、関数記録領域がスタック領域をオーバーフローしてプログラムの実行に支障を来す可能性を低減し、また関数記録領域の積み上げ及び破棄に要する処理負荷を低減して、全体の実行速度を向上させることが可能である。

【００１３】

第２発明に係る関数実行方法は、第１発明において、前記所定の条件は、被呼出関数の実行結果が呼出関数の実行結果となることであることを特徴とする。

【００１４】

第２発明に係る関数実行方法では、呼出関数による呼び出しが末尾呼び出し、即ち被呼出関数の実行結果が呼出関数の実行結果となり、被呼出関数の実行完了後、被呼出関数の関数記録領域に続いて、呼出関数の関数記録領域も破棄される呼び出し、例えばＪａｖａ言語の場合では被呼出関数を呼び出す命令の直後に任意個のプログラムカウンタ以外を変化させない命令を挟んで復帰命令があり、被呼出関数の戻り値の型と復帰命令の型とが一致し、被呼出関数を呼び出す命令から復帰命令の間に例外ハンドラが設定されていないという基準を満足することを条件とすることにより、呼出関数の関数記録領域を被呼出関数の関数記録領域として問題を発生することなく再利用することが可能である。

【００１５】

第３発明に係る関数実行方法は、第１発明又は第２発明において、前記呼出関数及び被呼出関数が異なるときに、被呼出関数の形式に基づいて、前記関数記録領域を変更することを特徴とする。

【００１６】

第３発明に係る関数実行方法では、呼出関数及び被呼出関数が異なるときに、呼出関数のために確保されている関数記録領域を被呼出関数の形式に基づいて変更することにより、被呼出関数を再利用した関数記録領域にて問題なく実行することが可能であり、また呼出関数及び被呼出関数が同じ関数であるときには、関数記録領域の形式を変更することなく、そのまま再利用するので形式の変更に要

する処理を無くし、全体としての処理速度を向上させることが可能である。

【００１７】

第４発明に係る関数実行装置は、メモリ中のスタック領域に、他の関数を呼び出す処理を含む呼出関数から呼び出された被呼出関数の形式に基づく関数記録領域を積み上げ、積み上げた関数記録領域を利用して被呼出関数を呼び出し、呼び出された被呼出関数の実行後、積み上げた関数記録領域を破棄する関数実行装置において、関数記録領域で実行される呼出関数を解析する手段と、該解析により呼出関数が所定の条件を満足すると判断した場合に、呼出関数を実行する前記関数記録領域を、被呼出関数を呼び出す領域として利用する手段とを備えることを特徴とする。

【００１８】

第４発明に係る関数実行装置では、呼出関数による呼び出しが、被呼出関数の実行結果を呼出関数の実行結果とする末尾呼び出しである等の所定の条件を満足すると判断した場合に、呼出関数の関数記録領域を被呼出関数の記録領域として再利用し、スタック領域に積み上げられる関数記録領域の数を低減することで、関数記録領域がスタック領域をオーバーフローしてプログラムの実行に支障を来す可能性を低減し、また関数記録領域の積み上げ及び破棄に要する処理負荷を低減して、全体の実行速度を向上させることが可能である。

【００１９】

第５発明に係るコンピュータプログラムは、コンピュータに、メモリ中のスタック領域に、他の関数を呼び出す処理を含む呼出関数から呼び出される被呼出関数の形式に基づく関数記録領域を積み上げさせ、積み上げた関数記録領域を利用して被呼出関数を呼び出させ、呼び出された被呼出関数の実行後、積み上げた関数記録領域を破棄させるコンピュータプログラムにおいて、コンピュータに、関数記録領域で実行される呼出関数を解析させる手順と、コンピュータに、解析により呼出関数が所定の条件を満足すると判断した場合に、呼出関数を実行する前記関数記録領域を、被呼出関数を呼び出す領域として利用させる手順とを含むことを特徴とする。

【００２０】

第５発明に係るコンピュータプログラムでは、携帯電話及びパーソナルコンピュータ等の処理装置を用いたＪＶＭにて実行することで、ＪＶＭが関数実行装置として動作することにより、呼出関数が所定の条件を満足した場合に、呼出関数の関数記録領域を被呼出関数の記録領域として再利用し、スタック領域に積み上げられる関数記録領域の数を低減することで、関数記録領域がスタック領域をオーバーフローしてプログラムの実行に支障を来す可能性を低減し、また関数記録領域の積み上げ及び破棄に要する処理負荷を低減して、全体の実行速度を向上させることが可能である。

【００２１】

第６発明に係るコンピュータプログラムは、第５発明において、前記所定の条件は、被呼出関数の実行結果が呼出関数の実行結果となることであることを特徴とする。

【００２２】

第６発明に係るコンピュータプログラムでは、呼出関数による呼び出しが末尾呼び出し、即ち被呼出関数の実行結果が呼出関数の実行結果となり、被呼出関数の実行完了後、被呼出関数の関数記録領域に続いて、呼出関数の関数記録領域も破棄される呼び出しであることを条件とすることにより、呼出関数の関数記録領域を被呼出関数の関数記録領域として問題を発生することなく再利用することが可能である。

【００２３】

第７発明に係るコンピュータプログラムは、第５発明又は第６発明において、前記呼出関数及び被呼出関数が異なるときに、被呼出関数の形式に基づいて、前記関数記録領域を変更することを特徴とする。

【００２４】

第７発明に係るコンピュータプログラムでは、呼出関数及び被呼出関数が異なる関数であるときに、呼出関数のために確保されている関数記録領域を被呼出関数の形式に基づいて変更することにより、被呼出関数を再利用した関数記録領域にて問題なく実行することが可能であり、また呼出関数及び被呼出関数が同じ関数であるときには、関数記録領域の形式を変更することなく、そのまま再利用す

るので形式の変更に要する処理を無くし、全体としての処理速度を向上させることが可能である。

【００２５】

第８発明に係るコンピュータプログラムは、コンピュータに、メモリ中のスタック領域に、他の関数を呼び出す処理を含む呼出関数から呼び出される被呼出関数の形式に基づく関数記録領域を積み上げさせ、積み上げた関数記録領域を利用して被呼出関数を呼び出させ、呼び出された被呼出関数の実行後、積み上げた関数記録領域を破棄させるコンピュータプログラムにおいて、コンピュータに、関数記録領域で実行される第１呼出関数を解析させる手順と、コンピュータに、解析により第１呼出関数が所定の条件を満足すると判断した場合に、呼出関数を実行する前記関数記録領域を、被呼出関数を呼び出す領域として利用させる処理を含む第１と異なる第２呼出関数を、第１呼出関数の代替関数として実行させる手順とを含むことを特徴とする。

【００２６】

第８発明に係るコンピュータプログラムでは、携帯電話及びパーソナルコンピュータ等の処理装置を用いたＪＶＭにて実行することで、ＪＶＭが関数実行装置として動作することにより、第１呼出関数による呼び出しが、被呼出関数の実行結果を呼出関数の実行結果とする末尾呼び出しである等の所定の条件を満足すると判断した場合に、利用した関数記録領域を被呼出関数の記録領域として再利用する処理を含む新たに用意された関数である第２呼出関数を、第１呼出関数の代替関数として実行し、これによりスタック領域に積み上げられる関数記録領域の数を低減して、関数記録領域がスタック領域をオーバーフローしてプログラムの実行に支障を来す可能性を低減し、また関数記録領域の積み上げ及び破棄に要する処理負荷を低減して、全体の実行速度を向上させることが可能である。

【００２７】

第９発明に係るコンピュータでの読み取りが可能な記録媒体は、コンピュータに、メモリ中のスタック領域に、他の関数を呼び出す処理を含む呼出関数から呼び出される被呼出関数の形式に基づく関数記録領域を積み上げさせ、積み上げた関数記録領域を利用して被呼出関数を呼び出させ、呼び出された被呼出関数の実

行後、積み上げた関数記録領域を破棄させるコンピュータプログラムを記録してある、コンピュータでの読み取りが可能な記録媒体において、コンピュータに、関数記録領域で実行される呼出関数を解析させる手順と、コンピュータに、解析により呼出関数が所定の条件を満足すると判断した場合に、呼出関数を実行する前記関数記録領域を、被呼出関数を呼び出す領域として利用させる手順と、を含むコンピュータプログラムを記録してあることを特徴とする。

【００２８】

第９発明に係るコンピュータでの読み取りが可能な記録媒体では、記録されているコンピュータプログラムを携帯電話及びパーソナルコンピュータ等の処理装置を用いたＪＶＭにて実行することで、ＪＶＭが関数実行装置として動作することにより、呼出関数が所定の条件を満足した場合に、呼出関数の関数記録領域を被呼出関数の記録領域として再利用し、スタック領域に積み上げられる関数記録領域の数を低減することで、関数記録領域がスタック領域をオーバーフローしてプログラムの実行に支障を来す可能性を低減し、また関数記録領域の積み上げ及び破棄に要する処理負荷を低減して、全体の実行速度を向上させることが可能である。

【００２９】

【発明の実施の形態】

以下、本発明をその実施の形態を示す図面に基づいて詳述する。

図１は本発明の関数実行装置を示すブロック図である。

図中１０は携帯電話及びパーソナルコンピュータ等の処理装置を用いた本発明の関数実行装置であり、関数実行装置１０は、本発明の関数実行装置用のコンピュータプログラム及びデータ等の情報を記録したＣＤ－ＲＯＭ及びメモリカード等の記録媒体ＲＥＣからコンピュータプログラム及びデータ等の情報を読み取る補助記憶手段１２、補助記憶手段１２により読み取られたコンピュータプログラム及びデータ等の情報を記録するハードディスク等の記録手段１３、並びに各種情報を一時的に記録するメモリ手段１４を備えている。

そして記録手段１３からコンピュータプログラム及びデータ等の情報を読み取り、メモリ手段１４に記録してＣＰＵ１１により実行することで、処理装置（コ

ンピュータ）は本発明の関数実行装置として動作する。

【００３０】

また関数実行装置１０は、モデム、ＴＡ(Terminal Adapter)、及びアンテナ等の通信手段１５を備えており、通信手段１５によりインターネット等の通信ネットワークＮＷに接続して、通信ネットワークＮＷに接続するウェブサーバコンピュータ等の記録装置２０が備える記録媒体２１に記録された本発明のコンピュータプログラム及びデータ等の情報を取り込み、実行するようにしてもよい。

【００３１】

次に本発明の関数実行方法の処理内容を説明する。

本発明の関数実行方法は、Ｊａｖａ言語等の言語にて記述され、ＪＶＭのバイトコードにコンパイルされた複数の命令からなる関数を複数含むプログラムを実行する方法に適用されるものであり、メモリ手段１４中のスタック領域に、実行すべき関数についての引数の個数及びローカル変数領域の大きさ等の形式に基づく関数記録領域を積み上げ、積み上げた関数記録領域に、関数を呼び出して実行する関数実行方法を基本とする。

【００３２】

図２は本発明の関数実行方法を概念的に示す説明図である。

図２（ａ）はスタック関数領域に確保された関数記録領域にて関数Ｆが実行されている状態を示す。

そして関数記録領域にて実行される関数Ｆを呼出関数とし、関数Ｆに呼び出される被呼出関数である関数Ｇを呼び出す場合、図２（ｂ）に示すように関数Ｆの関数記録領域に、関数Ｇを呼び出すための関数記録領域を新たに積み上げ、積み上げた関数記録領域を利用して関数Ｇを呼び出し実行する。

【００３３】

さらに関数Ｇを呼出関数とし、関数Ｇに呼び出される被呼出関数である関数Ｈを呼び出す場合で、関数Ｈの実行結果が関数Ｇの実行結果となる末尾呼出であるときに、図２（ｃ）に示すように関数Ｈのための関数記録領域を新たに積み上げることなく、関数Ｇが記録されている関数記録領域を、関数Ｈを呼び出すための関数記録領域とする。

そして関数Ｈの実行結果は関数Ｆに渡され、関数Ｈを実行した関数記録領域は破棄される。

【００３４】

次に本発明の関数実行方法における解析処理を図３に示すフローチャートを用いて説明する。

ＪＶＭ等のバイトコードによる関数を実行するにあたり、先ず新たに関数記録領域を確保し、そして呼び出すべき関数が、ｉｎｖｏｋｅｓｔａｔｉｃ、ｉｎｖｏｋｅｖｉｒｔｕａｌ、ｉｎｖｏｋｅｓｐｅｃｉａｌ、及びｉｎｖｏｋｅｉｎｔｅｒｆａｃｅ等の被呼出関数を呼び出す命令を含む呼出関数である場合に、当該呼出関数を解析し（Ｓ１０１）、解析により、呼出関数が、被呼出関数の実行結果が呼出関数の実行結果となるという条件を満足する末尾呼出関数であると判断したときに（Ｓ１０２：Ｙ）、更に呼出関数及び被呼出関数が同じであるか否かを判別する（Ｓ１０３）。

なおステップＳ１０２において末尾呼出関数であると判断する基準のＪａｖａ言語における具体例としては、被呼出関数を呼び出す命令の直後に任意個のプログラムカウンタ以外を変化させないｎｏｐ及びｇｏｔｏ等の命令を挟んで復帰命令があり、被呼出関数の戻り値の型と、ｉｒｅｔｕｒｎ、ｌｒｅｔｕｒｎ、ｆｒｅｔｕｒｎ、ｄｒｅｔｕｒｎ、ａｒｅｔｕｒｎ、及びｒｅｔｕｒｎ等の復帰命令の型とが一致し、被呼出関数を呼び出す命令から復帰命令の間に例外ハンドラが設定されていない等の基準がある。

【００３５】

ステップＳ１０３において、呼出関数及び被呼出関数が異なると判別したとき（Ｓ１０３：Ｎ）、当該呼出関数（第１呼出関数）を、被呼出関数を呼び出す命令を予め用意している代替命令に置換した再帰代替関数（第２呼出関数）に置換する（Ｓ１０４）。

呼出関数及び被呼出関数が同じであると判別したとき（Ｓ１０３：Ｙ）当該呼出関数（第１呼出関数）を、呼出関数と同じである被呼出関数を呼び出す命令を予め用意している代替命令に置換した自己再帰代替関数（第２呼出関数）に置換する（Ｓ１０５）。

なお置換すべき命令が複数個含まれる場合、該当する全ての命令を置換した関数に置換される。

ステップＳ１０２において、末尾呼出関数でないと判断したとき（Ｓ１０２：Ｎ）、ステップＳ１０３～Ｓ１０５の処理は行われない。

【００３６】

なお呼出関数の代替関数として用いられるステップＳ１０４に示す再帰代替関数及びステップＳ１０５に示す自己再帰代替関数として、以下に示す新たな命令を含む関数を用意しておく。

即ち、

ｉｎｖｏｋｅｓｔａｔｉｃ、

ｉｎｖｏｋｅｖｉｒｔｕａｌ、

ｉｎｖｏｋｅｓｐｅｃｉａｌ、

及びｉｎｖｏｋｅｉｎｔｅｒｆａｃｅ

等の呼出命令を含む呼出関数の再帰代替関数として、

ｔａｉｌｉｎｖｏｋｅｓｔａｔｉｃ、

ｔａｉｌｉｎｖｏｋｅｖｉｒｔｕａｌ、

ｔａｉｌｉｎｖｏｋｅｓｐｅｃｉａｌ、

及びｔａｉｌｉｎｖｏｋｅｉｎｔｅｒｆａｃｅ

等の代替命令を用意し、用意した代替命令を含む再帰代替関数を用いる。

また自己再帰代替関数として、

ｓｅｌｆｔａｉｌｉｎｖｏｋｅｓｔａｔｉｃ、

ｓｅｌｆｔａｉｌｉｎｖｏｋｅｖｉｒｔｕａｌ、

ｓｅｌｆｔａｉｌｉｎｖｏｋｅｓｐｅｃｉａｌ、

及びｓｅｌｆｔａｉｌｉｎｖｏｋｅｉｎｔｅｒｆａｃｅ

等の代替命令を用意し、用意した代替命令を含む自己再帰代替関数を用いる。

これらの命令を含む関数は、解析処理では置換されるだけであり、実際に実行されるのは、置換された関数を実行する実行処理においてであるので、以下の実行処理の説明にてその機能を説明する。

【００３７】

次に本発明の関数実行方法における実行処理を図４及び図５に示すフローチャートを用いて説明する。

解析処理により必要に応じて置換がなされた関数を実行する場合、先ず実行すべき呼出関数が、代替関数である再帰代替関数或いは自己再帰代替関数、又は代替関数で無いかを判別し（Ｓ２０１）、再帰代替関数であると判断した場合（Ｓ２０１：１）、再帰代替関数（第２呼出関数）の形式に基づく関数記録領域を積み上げ（Ｓ２０２）、積み上げた関数記録領域を利用して再帰代替関数を呼び出し（Ｓ２０３）、ステップＳ２０２にて積み上げた関数記録領域にて再帰代替関数を実行する（Ｓ２０４）

【００３８】

そして代替関数に含まれるｔａｉｌｉｎｖｏｋｅｓｔａｔｉｃ等の代替命令による処理により、呼出関数（第１呼出関数）として実行している再帰代替関数（第２呼出関数）から被呼出関数を呼び出す場合に、新たに関数記録領域を積み上げることなく、再帰代替関数を実行した関数記録領域を、被呼出関数の形式に基づいて変更し（Ｓ２０５）、形式を変更した関数記録領域を、被呼出関数を呼び出す領域として利用して（Ｓ２０６）、被呼出関数を呼び出し（Ｓ２０７）、呼び出した被呼出関数を実行し（Ｓ２０８）、被呼出関数の実行後、被呼出関数の実行に利用した関数記録領域を破棄する（Ｓ２０９）。

【００３９】

ステップＳ２０１において、自己再帰代替関数であると判断した場合（Ｓ２０１：２）、自己再帰代替関数（第２呼出関数）の形式に基づく関数記録領域を積み上げ（Ｓ２１０）、積み上げた関数記録領域を利用して自己再帰代替関数を呼び出し（Ｓ２１１）、ステップ２１０にて積み上げた関数記録領域にて自己再帰代替関数を実行する（Ｓ２１２）。

【００４０】

そして自己再帰代替関数に含まれるｓｅｌｆｔａｉｌｉｎｖｏｋｅｓｔａｔｉｃ等の代替命令による処理により、呼出関数（第１呼出関数）として実行している自己再帰代替関数（第２呼出関数）から被呼出関数を呼び出す場合に、新たに関数記録領域を積み上げることなく、自己再帰代替関数を実行した関数記録領域

を、被呼出関数を呼び出す領域として利用して（Ｓ２１３）、被呼出関数を呼び出し（Ｓ２１４）、呼び出した被呼出関数を実行し（Ｓ２１５）、被呼出関数の実行後、被呼出関数の実行に利用した関数記録領域を破棄する（Ｓ２１６）。

【００４１】

ステップＳ２０１において、代替関数でないと判断した場合（Ｓ２０１：３）、呼出関数の形式に基づく関数記録領域を積み上げ（Ｓ２１７）、積み上げた関数記録領域を利用して呼出関数を呼び出し（Ｓ２１８）、ステップ２１７にて積み上げられた関数記録領域にて呼出関数を実行する（Ｓ２１９）。

【００４２】

そして呼出関数から被呼出関数を呼び出す場合に、被呼出関数の形式に基づく新たな関数記録領域を積み上げ（Ｓ２２０）、積み上げた関数記録領域を利用して被呼出関数を呼び出し（Ｓ２２１）、ステップＳ２２０にて積み上げた関数記録領域にて被呼出関数を実行し（Ｓ２２２）、被呼出関数の実行後、被呼出関数の実行に利用した関数記録領域を破棄し（Ｓ２２３）、更に呼出関数における被呼出関数実行後の処理を実行して、呼出関数の実行に利用した関数記録領域を破棄する（Ｓ２２４）。

【００４３】

次に本発明の関数実行方法及び従来の関数実行方法の処理速度を比較した結果を図６に示すグラフを用いて説明する。

図６では横軸に末尾呼出関数の呼び出し深さをとり、縦軸にマイクロ秒（ｕｓ）を単位とする処理時間をとって、その関係を示したものであり、×印は呼出関数を呼び出す都度、関数記録領域を積み上げる従来の関数実行方法による呼び出し深さと処理時間との関係を示し、□印は呼出関数が末尾呼出の場合に関数記録領域を再利用する本発明の関数実行方法による呼び出し深さと処理時間との関係を示し、○印は呼出関数と被呼出関数とが同じである自己末尾呼出関数を扱う場合の呼び出し深さと処理時間との関係を示している。

図６に示すように従来の関数実行方法と比較して、本発明の関数実行方法は処理時間が短く、特に自己末尾呼出関数ではその傾向が顕著である。

【００４４】

また本発明の実施の形態としては、ＪＩＴ(Just In Time Compiler)と呼ばれるＪＶＭの実装技術を用いてもよく、ＪＩＴ技術により、ＪＶＭのバイトコードを機械語に変換（コンパイル）して、ＣＰＵ１１にて実行することで、実行処理の速度を向上させることが可能である。

【００４５】

そして前記実施の形態では、呼出関数が末尾呼出又は自己末尾呼出である場合に、代替関数を用いる形態を示したが、本発明はこれに限らず、関数実行時に都度再帰判定を行い、末尾再帰であると判断した場合に、末尾再帰処理を行うようにしてもよい。

【００４６】

【発明の効果】

以上詳述した如く本発明に係る命令実行方法、命令実行装置、コンピュータプログラム、及び記録媒体では、Ｊａｖａ言語等の言語にて記述され複数の関数を含むプログラムを、携帯電話及びパーソナルコンピュータ等の処理装置を用いたＪＶＭにて実行する場合で、メモリ中のスタック領域に、他の関数を呼び出す処理を呼出関数による呼び出しが、被呼出関数の実行結果を呼出関数の実行結果とする末尾呼び出しである等の所定の条件を満足すると判断したときに、呼出関数の関数記録領域を被呼出関数の記録領域として再利用し、スタック領域に積み上げられる関数記録領域の数を低減することで、関数記録領域がスタック領域をオーバーフローしてプログラムの実行に支障を来す可能性を低減し、また関数記録領域の積み上げ及び破棄に要する処理負荷を低減して、全体の実行速度を向上させることが可能である等、優れた効果を奏する。

【００４７】

また本発明では呼出関数及び被呼出関数が異なるときに、呼出関数のために確保されている関数記録領域を被呼出関数の形式に基づいて変更することにより、被呼出関数を再利用した関数記録領域にて問題なく実行することが可能であり、また呼出関数及び被呼出関数が同じ関数であるときには、関数記録領域の形式を変更することなく、そのまま再利用するので形式の変更に要する処理を無くし、全体としての処理速度を向上させることが可能である等、優れた効果を奏する。

【図面の簡単な説明】

【図１】

本発明の関数実行装置を示すブロック図である。

【図２】

本発明の関数実行方法を概念的に示す説明図である。

【図３】

本発明の関数実行方法における解析処理を示すフローチャートである。

【図４】

本発明の関数実行方法における実行処理を示すフローチャートである。

【図５】

本発明の関数実行方法における実行処理を示すフローチャートである。

【図６】

本発明の関数実行方法及び従来の関数実行方法の処理速度を示すグラフである。

【図７】

プログラムの実行に必要な関数を記録する関数記録領域を示す概念図である。

【符号の説明】

１０　関数実行装置

１１　ＣＰＵ

１２　補助記憶手段

１３　記録手段

１４　メモリ手段

１５　通信手段

２０　記録装置

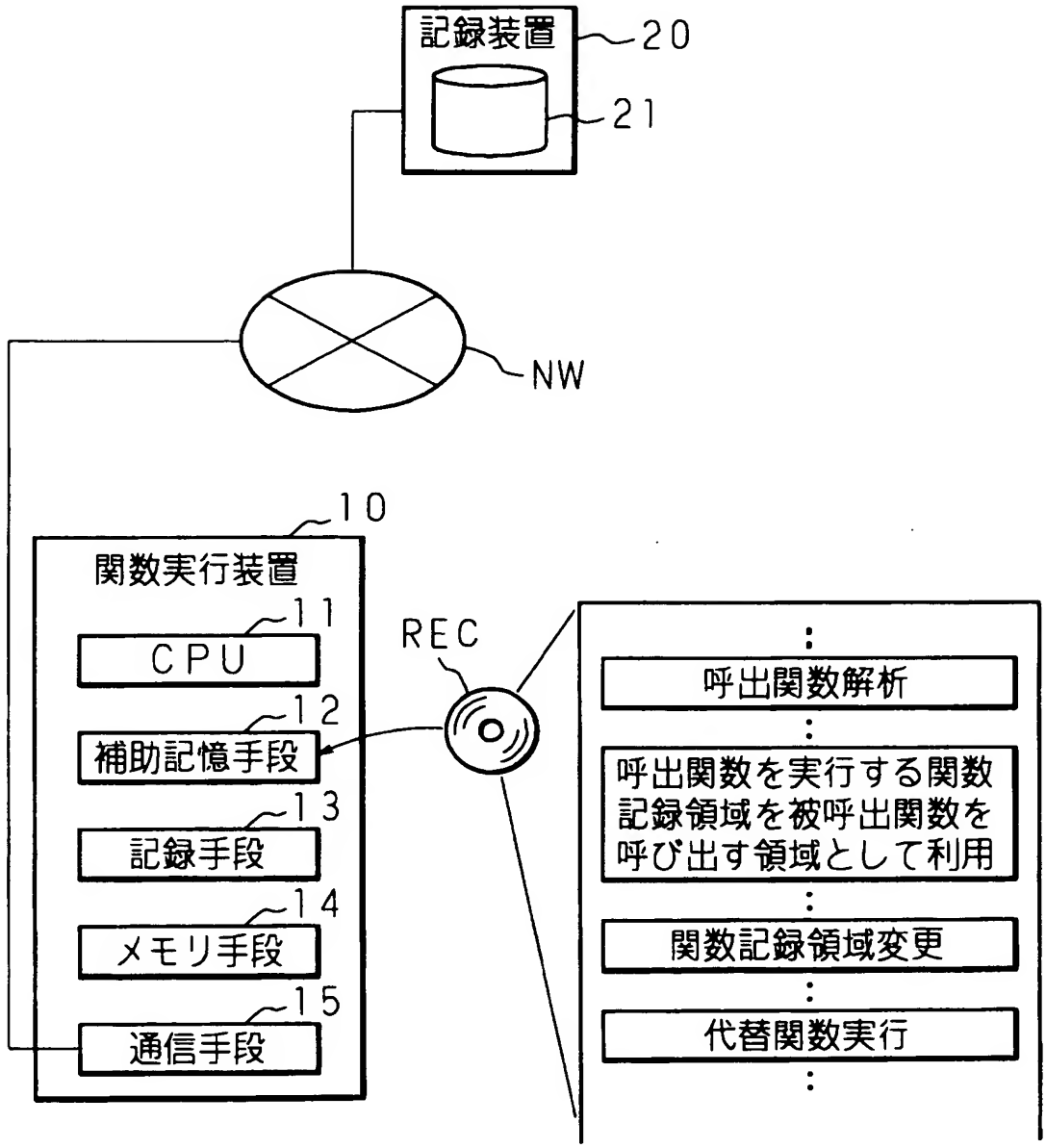
２１　記録媒体

ＲＥＣ　記録媒体

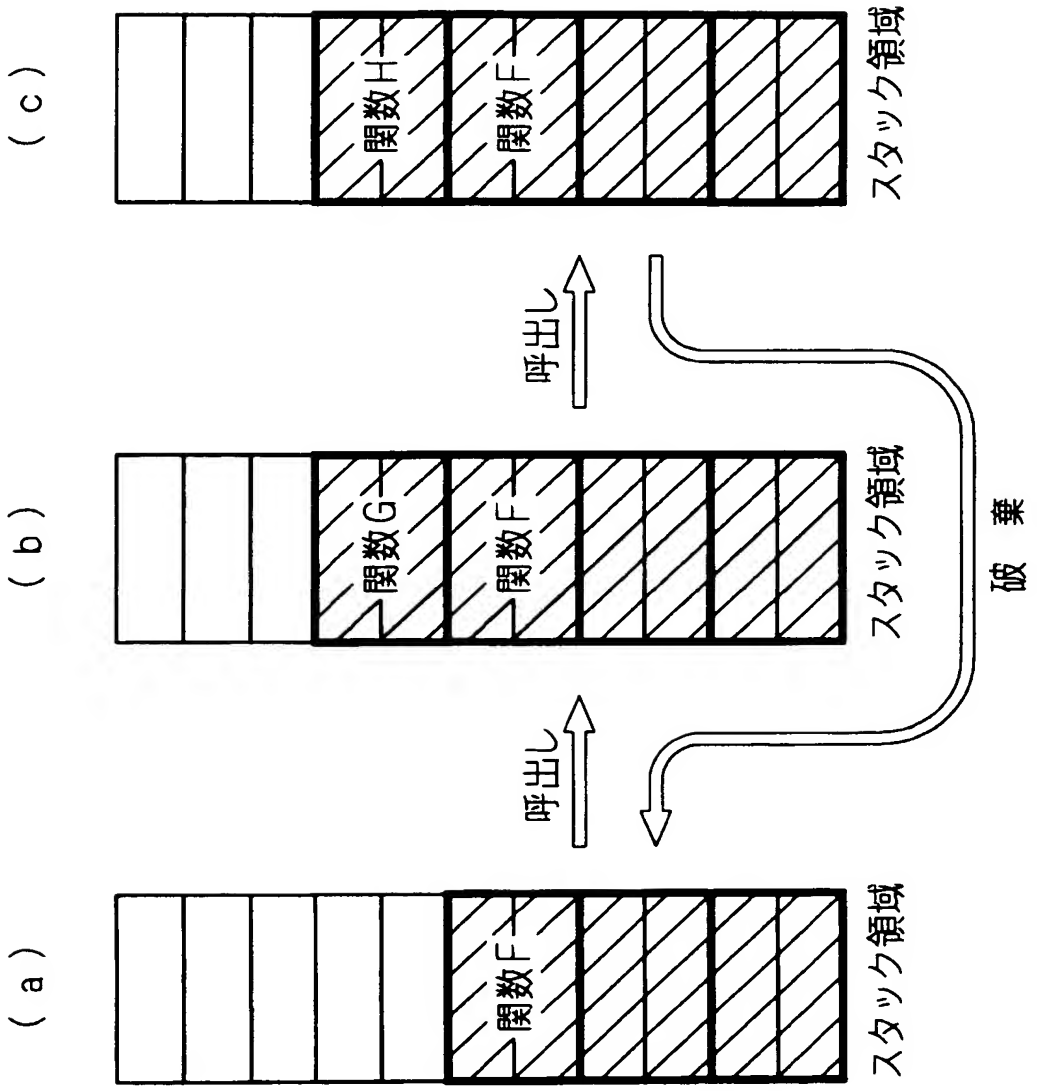
ＮＷ　通信ネットワーク

【書類名】　図面

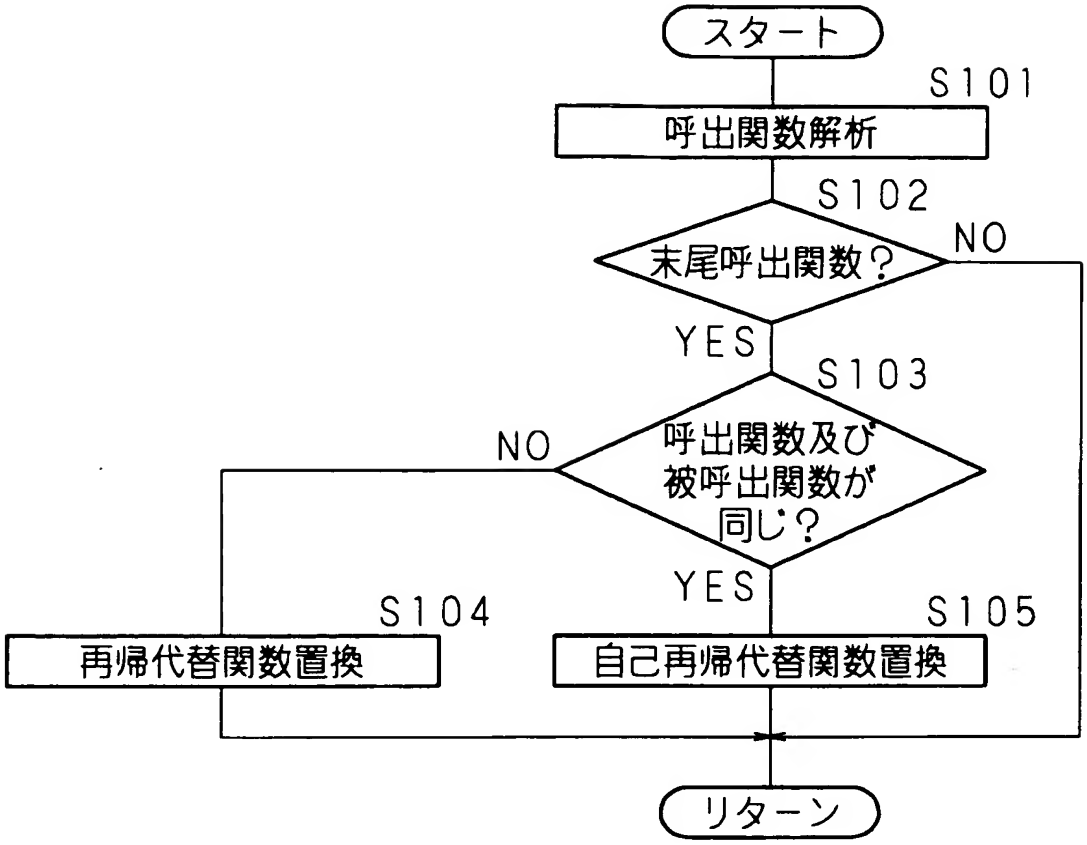
【図１】


記録装置
20
21
NW
10
関数実行装置
11
CPU
12
補助記憶手段
13
記録手段
14
メモリ手段
15
通信手段
REC
呼出関数解析
呼出関数を実行する関数記録領域を被呼出関数を呼び出す領域として利用
関数記録領域変更
代替関数実行

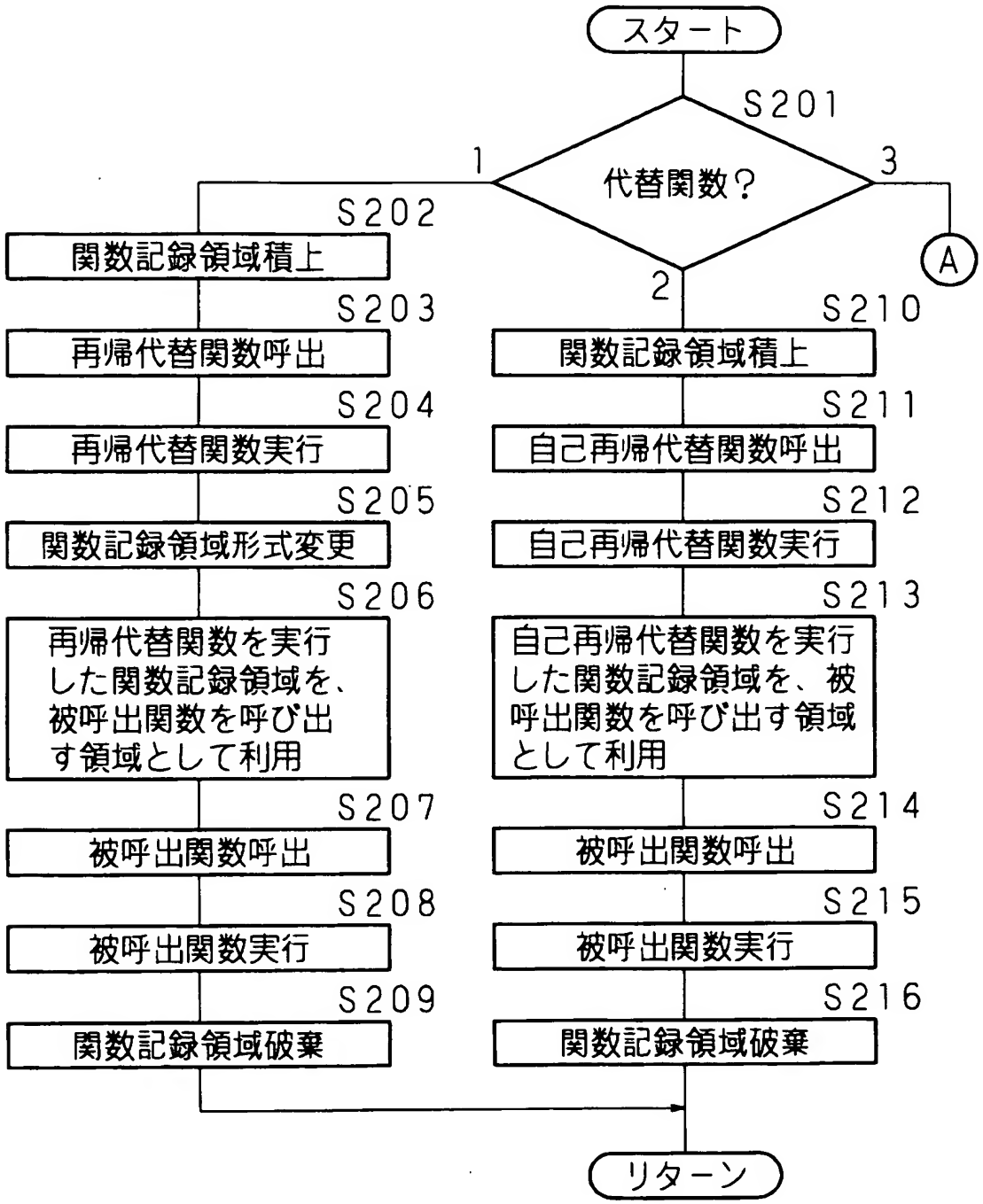
【図２】



【図３】


スタート
S101
呼出関数解析
S102
末尾呼出関数？
NO
YES
S103
呼出関数及び
被呼出関数が
同じ？
NO
YES
S104
再帰代替関数置換
S105
自己再帰代替関数置換
リターン

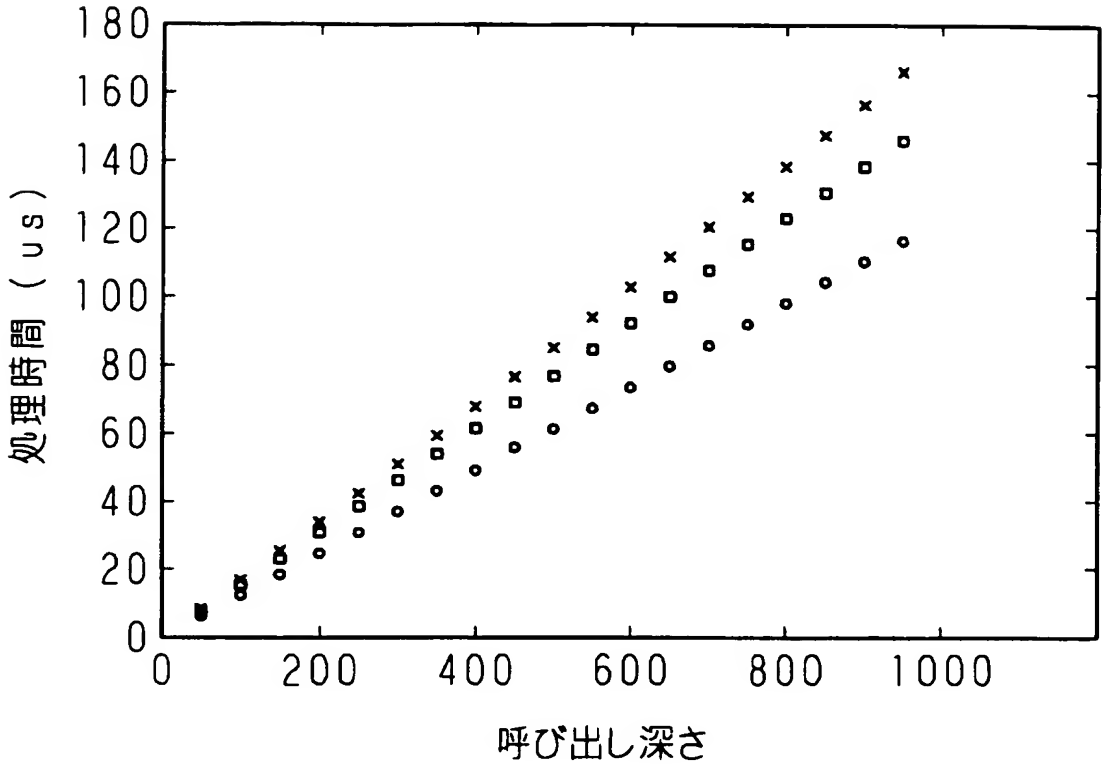
【図４】


スタート
S201
代替関数？
1
2
3
A
S202
関数記録領域積上
S203
再帰代替関数呼出
S204
再帰代替関数実行
S205
関数記録領域形式変更
S206
再帰代替関数を実行した関数記録領域を、被呼出関数を呼び出す領域として利用
S207
被呼出関数呼出
S208
被呼出関数実行
S209
関数記録領域破棄
S210
関数記録領域積上
S211
自己再帰代替関数呼出
S212
自己再帰代替関数実行
S213
自己再帰代替関数を実行した関数記録領域を、被呼出関数を呼び出す領域として利用
S214
被呼出関数呼出
S215
被呼出関数実行
S216
関数記録領域破棄
リターン

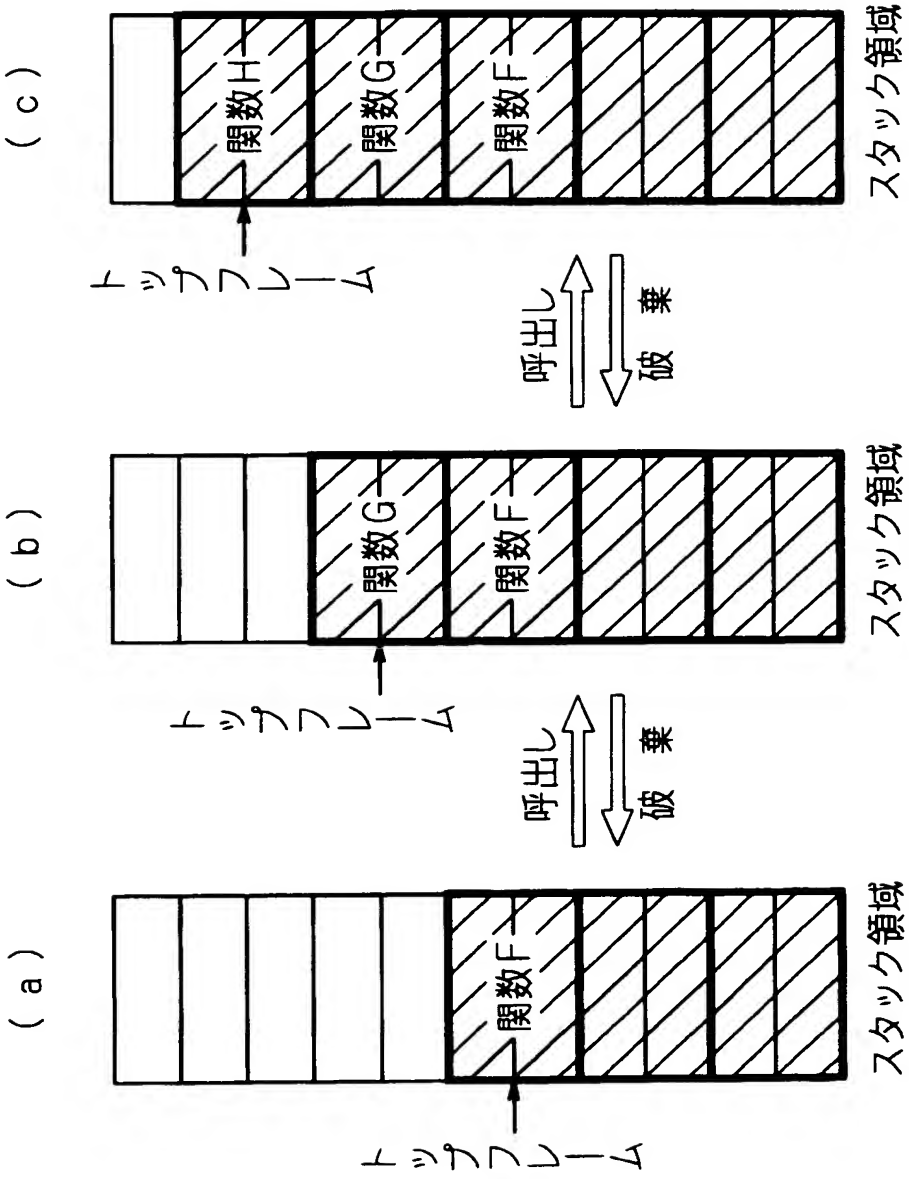
【図５】


A
S217
関数記録領域積上
S218
呼出関数呼出
S219
呼出関数実行
S220
新たな関数記録領域積上
S221
被呼出関数呼出
S222
被呼出関数実行
S223
関数記録領域破棄
S224
関数記録領域破棄
リターン

【図６】


処理時間（ｕｓ）
180
160
140
120
100
80
60
40
20
0
0
200
400
600
800
1000
呼び出し深さ

【図７】


（ｃ）
関数H
関数G
関数F
スタック領域
トップフレーム
呼出し
破棄
（ｂ）
関数G
関数F
スタック領域
トップフレーム
呼出し
破棄
（ａ）
関数F
スタック領域
トップフレーム

【書類名】　要約書

【要約】

【課題】　メモリ中のスタック領域に、他の関数を呼び出す処理を含む呼出関数から呼び出される被呼出関数の形式に基づく関数記録領域を積み上げて、被呼出関数を呼び出し、被呼出関数の実行後、積み上げた関数記録領域を破棄するプログラムの実行において、関数記録領域のオーバーフローを防止し、全体の実行速度を向上させる関数実行方法、関数実行装置、コンピュータプログラム、及び記録媒体を提供する。

【解決手段】　ＪＶＭ等のバイトコードによる呼出関数を解析し、呼出関数が末尾呼出関数であると判断したときに、呼出関数を所定の代替関数に置換し、置換した代替関数を実行することで、代替関数である呼出関数から被呼出関数を呼び出す場合に、呼出関数の実行に利用した関数記録領域を利用して被呼出関数を呼び出し、呼び出した被呼出関数を実行する。

【選択図】　図２

特願２００１－０８８８５０

出　願　人　履　歴　情　報

識別番号　　［８９９０００００４６］

１．変更年月日　　１９９９年　９月１７日
［変更理由］　新規登録
住　所　京都府京都市下京区中堂寺粟田町１番地
氏　名　関西ティー・エル・オー株式会社

２．変更年月日　　２００２年　８月　２日
［変更理由］　住所変更
住　所　京都府京都市下京区中堂寺粟田町９３番地
氏　名　関西ティー・エル・オー株式会社